



MARKO JUNG

GALACTIC VICEROY OF RESEARCH EXCELLENCE

 m@mju.ng

 [@mjung](https://twitter.com/mjung)

 [fb.com/markohjung](https://facebook.com/markohjung)

OXFORD AT A GLANCE



22,000+ students

17,000 employees

£m 1,429.3 turn-over (£m 12.3 deficit)

Highly federated organisation with independent and self-governing members:

- 80+ departments & units
- 38 colleges and 6 permanent private halls

IT AT OXFORD



Central IT Services department

- Core infrastructure services & projects
- IT learning programme
- Value added services

Local IT Support in the departments, units, and colleges

→ Neither technical nor organisational consistency but that there will be **individuality, independence and ingenuity in abundance**, characteristics that the University of Oxford is renowned for and has always encouraged.



DESKTOP SERVICES

INFRASTRUCTURE PROVIDER



BUILD THE BEST DEVICE MANAGEMENT TOOLSET



**RELIABILITY, AVAILABILITY &
SECURITY MULTI TENANCY
INTERFACES FOR DEVOPS,
SUPPORT & END USERS
OPEN DEVELOPMENT MODEL
KNOWLEDGE TRANSFER
COLLABORATION SHARING**



DEPLOYMENT WORKFLOWS



NetBoot & Imaging



Enrolment



DEP / MDM

DEP IN A FEDERATED ENVIRONMENT



- Individual DEP account per Oxford unit
 - distributed purchasing
 - federated control of DEP assignments
- Managed in corresponding JSS sites

Orchard **DEPLOYMENT MODEL**



Netboot &
Imaging



Apple DEP



MDM &
Config



Directory
Binding



Branding &
Custom Tools



Updates



Apps











App Store



Ongoing
Maintenance

SOFTWARE DEPLOYMENT

SOFTWARE STAGING STRATEGY

Week 1	 41.0		 40.0.3
Week 2		 41.0	 40.0.3
Week 3	 42.0β1		 41.0
Week 4	Unstable	 42.0β1 Testing	 41.0 Stable

DEPRECATED JSS MODEL

Installer Policy + Installer SmartGroup

Updates:



- Stable Update Policy +
Stable Update SmartGroup
- Testing Update Policy +
Testing Update SmartGroup
- Unstable Update Policy +
Unstable Update SmartGroup

→ At least 11 JSS Objects per software title

ALTERNATIVE APPROACHES



- Patchoo! (Lachlan Stewart)

<https://github.com/patchoo/patchoo/>

*"Why not just use munki?
That's certainly an option, and
many people do use munki and
Casper. The tools are designed
differently and it's hard to draw
parallels between their workflows.
It would be great to leverage
munki, but currently it is difficult to
integrate what's exposed via the
JSS API and munki."*



ALTERNATIVE APPROACHES



- jss-autopkg-addon (Allister Banks)

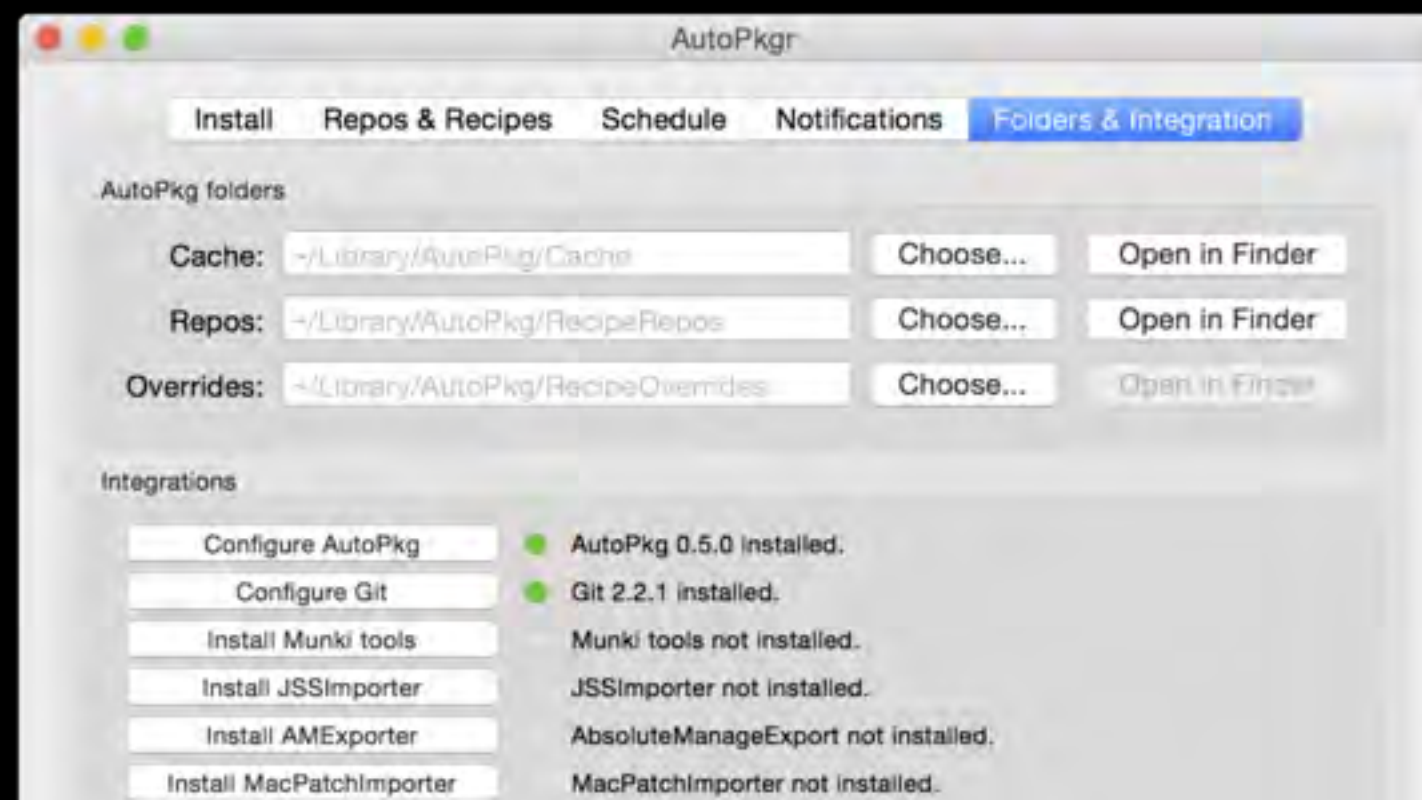
<https://github.com/arubdesu/jss-autopkg-addon>

- autopkg JSS importer (Shea Craig)

<https://github.com/sheagcraig/JSSImporter>

- AutoPkgr (Linde Group)

<http://www.lindegroup.com/autopkgr>



AUTOPKG



- Automated preparation of software for managed distribution
- Community maintained recipes (PropertyList XML) to automate complex tasks

Firefox.download.recipe

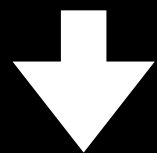
Firefox.pkg.recipe

Firefox.munki.recipe

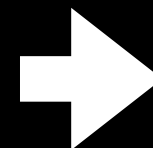
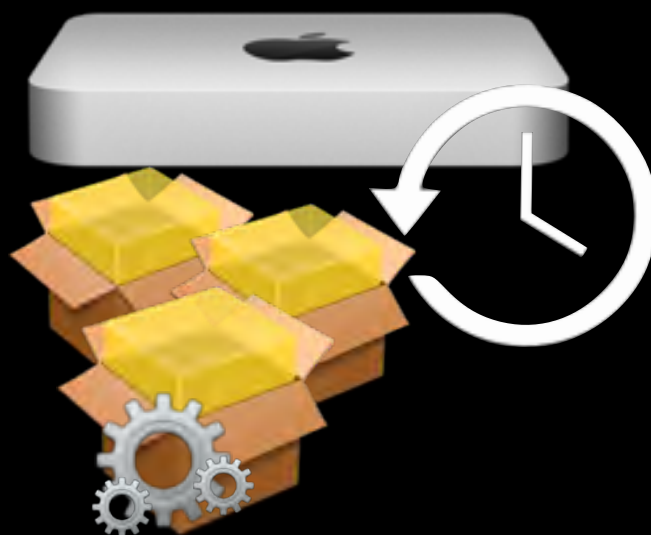
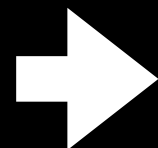
- Excellent integration with Munki
- Workflows for other management tools like Absolute Manage, JAMF Casper Suite
- MacSysadmin 2014– G. Neagle, T. Sutton
AUTOPKG: CROWD-SOURCING MAC PACKAGING AND DEPLOYMENT

<http://docs.macsysadmin.se/2014/2014doc.html>

AUTOPKG WORKFLOW



GitHub



RECIPE DEVELOPMENT
LOCAL WORKSTATIONS

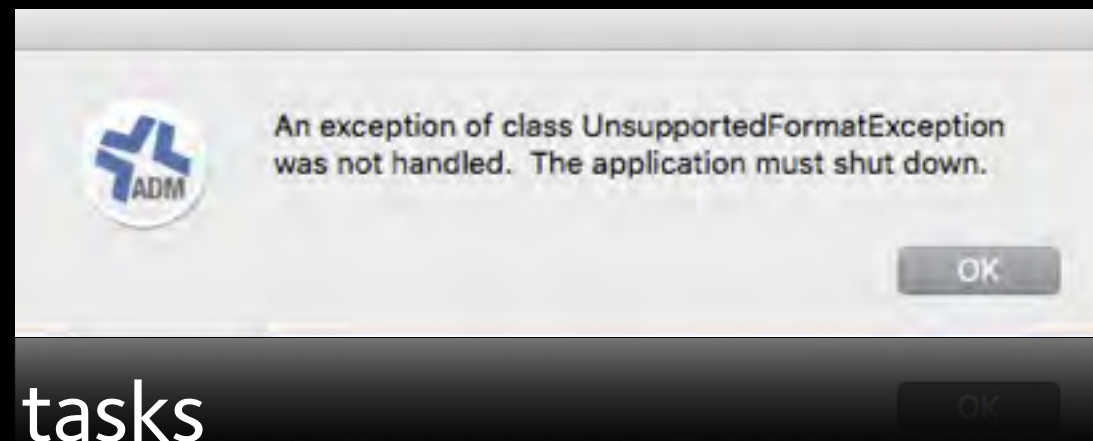
AUTOPKG
BUILD HOST

HOW TO DEPLOY
USING THE JSS?

JSS PACKAGE MANAGEMENT



- JSS package management at a glance:
 - Packages cumbersome to add to JSS
 - No concept of updates
 - Text comparison for versions
 - Not much automation of common tasks (e.g. running Apps, dependencies)
 - Packages (and Scripts) are global JSS objects
- Dated approach lacking automation and collaboration features



MODERN PACKAGE MANAGEMENT



European Macintosh System Administrators Meeting, Sweden, September 2014

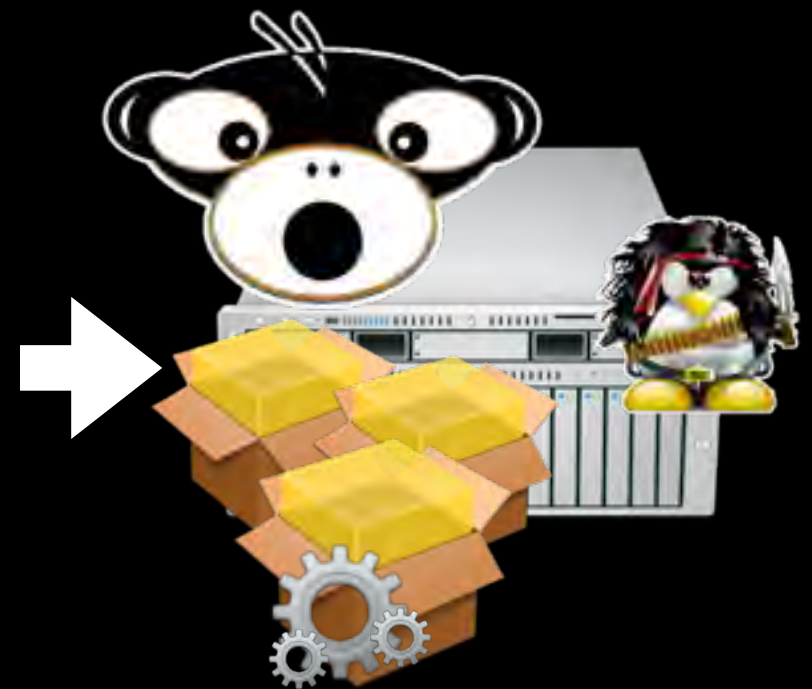
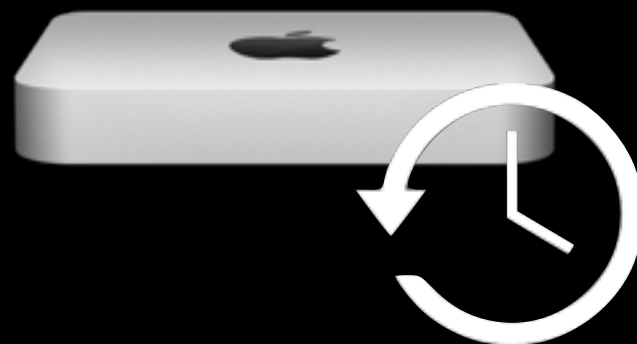
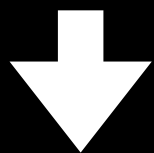
MUNKI 2



- System administrator friendly toolset
 - text based configuration
 - powerful command line tools
- Familiar user interface:
Managed Software Center.app
- Excellent ecosystem of tools
- All features JSS is missing
- Open Source, Apache 2 licensed
- MacSysadmin 2014– G. Neagle: *WHAT'S NEW WITH MUNKI?*
<http://docs.macsysadmin.se/2014/2014doc.html>



AUTOPKG WORKFLOW



RECIPE DEVELOPMENT
LOCAL WORKSTATIONS

AUTOPKG
BUILD HOST

MUNKI REPOSITORY
DATA CENTRES

Orchard

Search Inventory >

Licensed Software >

Policies >

Configuration Profiles >

Managed Preferences >

Restricted Software >

PreStage Imaging >

Mac App Store Apps >

eBooks >

Smart Computer Groups >

Static Computer Groups >

Enrollment Invitations >

PreStage Enrollments >

Management Settings >

JAMF

software

© 2008-2015 JAMF Software, LLC

Computers

Mobile Devices

Users

Full JSS ▼

Marko

Policies (1530)

+ New

Show All

Hide All

Filter Policies

Name	Frequency	Trigger	Scope	Site
+ Communications and Web				
- Core Software				
+ cocoaDialog	Once per computer	BuildPre	Software: Core (all OS)	None
+ Deploy Weekly Maintenance	Once per computer	Startup, Check-in, BuildPost, Self Service	All computers	None
+ dockutil	Once per computer	SoftwareInstall	Software: Core (all OS)	None
+ HP Printer Drivers	Once per computer		Software: Core (all OS)	None
+ MMP Version	Once per computer	BuildPre	All computers	None
+ Orchard Branding	Once per computer	BuildPre	Software: Core (all OS)	None
+ Planting 100: Enforce Orchard Management Account	Ongoing	Planting	All computers	None
+ Planting 200: Main Setup Tasks	Ongoing	Planting	All computers	None
+ Ricoh Printer Drivers	Once per computer	Self Service	Software: Core (all OS)	None
+ Sassafras Keyserver 6	Once per computer	Self Service	Software: Sassafras Keyserver (IT-Services) Install	None
+ Sophos Anti-Virus	Once per computer	SoftwareInstall, Self Service	Software: Core (all OS)	None

Orchard

Search Inventory >

Licensed Software >

Policies >

Configuration Profiles >

Managed Preferences >

Restricted Software >

PreStage Imaging >

Mac App Store Apps >

eBooks >

Smart Computer Groups >

Static Computer Groups >

Enrollment Invitations >

PreStage Enrollments >

Management Settings >

JAMF software
© 2002-2015 JAMF Software, LLC.

Computers

Mobile Devices

Users

Full JSS ▾

Marko

⌵

⚙

Policies (358)

+ New






⌵

⌶

Show All

Hide All

Filter Policies

Name	Frequency	Trigger	Scope	Site
Core Software				
+  <u>Deploy Weekly Maintenance</u>	Once per computer	Startup, Check-in, BuildPost, Self Service	All computers	None
+  <u>Orchard Branding</u>	Once per computer	BuildPre	Software: Core (all OS)	None
 <u>Orchard Software Centre</u>	Once per computer	Check-in, Self Service	No Orchard Software Centre	None
+  <u>Planting 100: Enforce Orchard Management Account</u>	Ongoing	Planting	All computers	None
+  <u>Planting 200: Main Setup Tasks</u>	Ongoing	Planting	All computers	None
+ Scripts				
+ Support				
+ System Administration				
+ Utilities				



AUTOPKG



TRELLO



MUNKI

AUTOPKG NIGHTLY BUILDS



```
#!/bin/bash
exec > >(logger -i -t autopkg-build) 2>&1

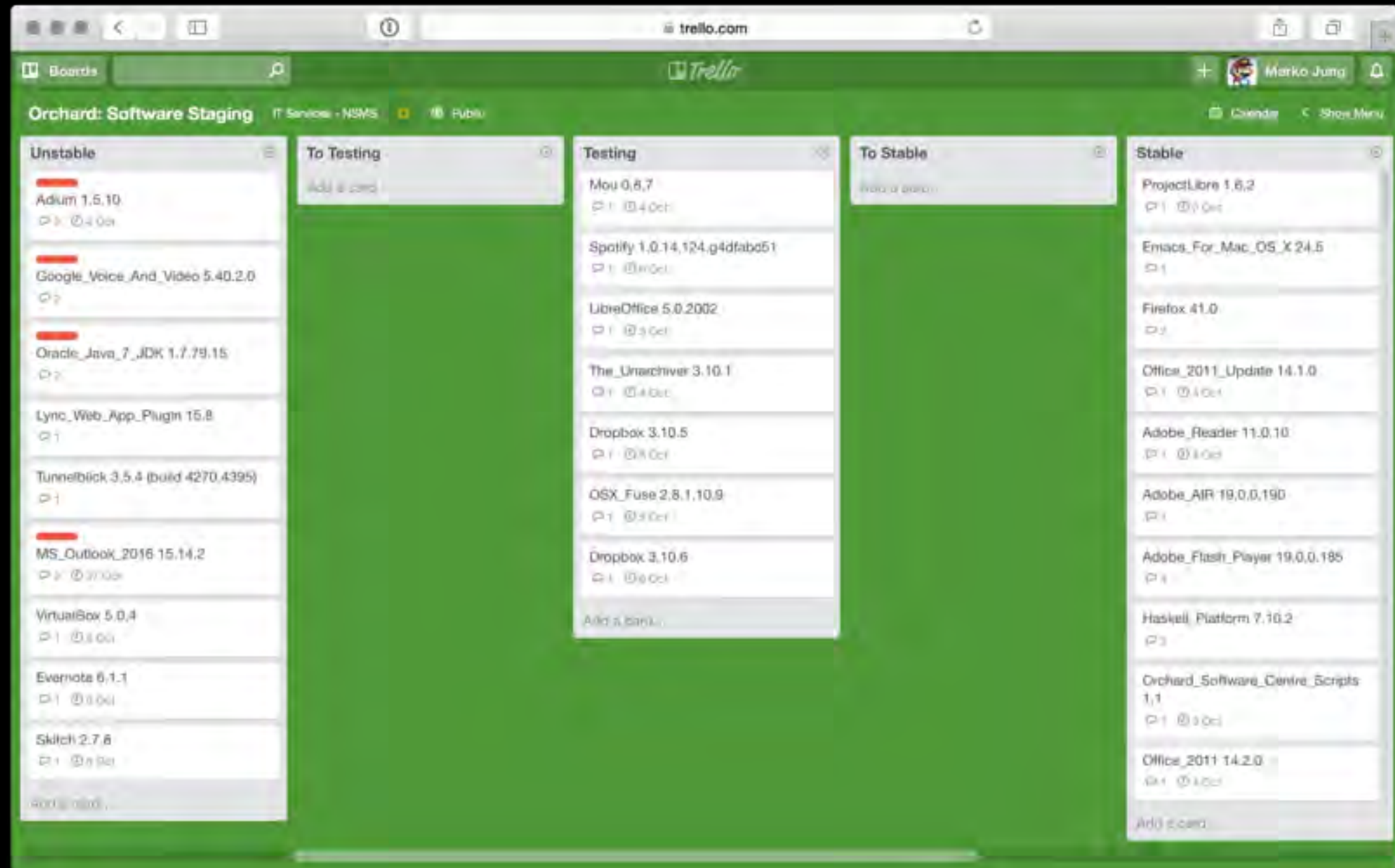
BUILD_RECIPE_DIR="/srv/autopkg/BuildRecipes"
EMAIL_ERRORS_TO='incidents@acme.corp'
my_name=$(basename $0)

set -o pipefail

echo "$(date) BEGIN AUTOPKG BUILD RUN"
for recipe in $(cd ${BUILD_RECIPE_DIR}; ls ); do
    OUTPUT=$(mktemp /tmp/${my_name}.XXXXXXXXXX)
    echo "$(date) Autopkg running ${recipe} ... "
    /usr/local/bin/autopkg run ${recipe} | tee ${OUTPUT}
    if [ $? -ne 0 ]; then
        mail -s "Failed to build ${recipe}" ${EMAIL_ERRORS_TO} < ${OUTPUT}
    fi
    echo "$(date) ... end run of $recipe"
    rm -f ${OUTPUT}
done
echo "$(date) END AUTOPKG BUILD RUN"
```



MUNKI-STAGING

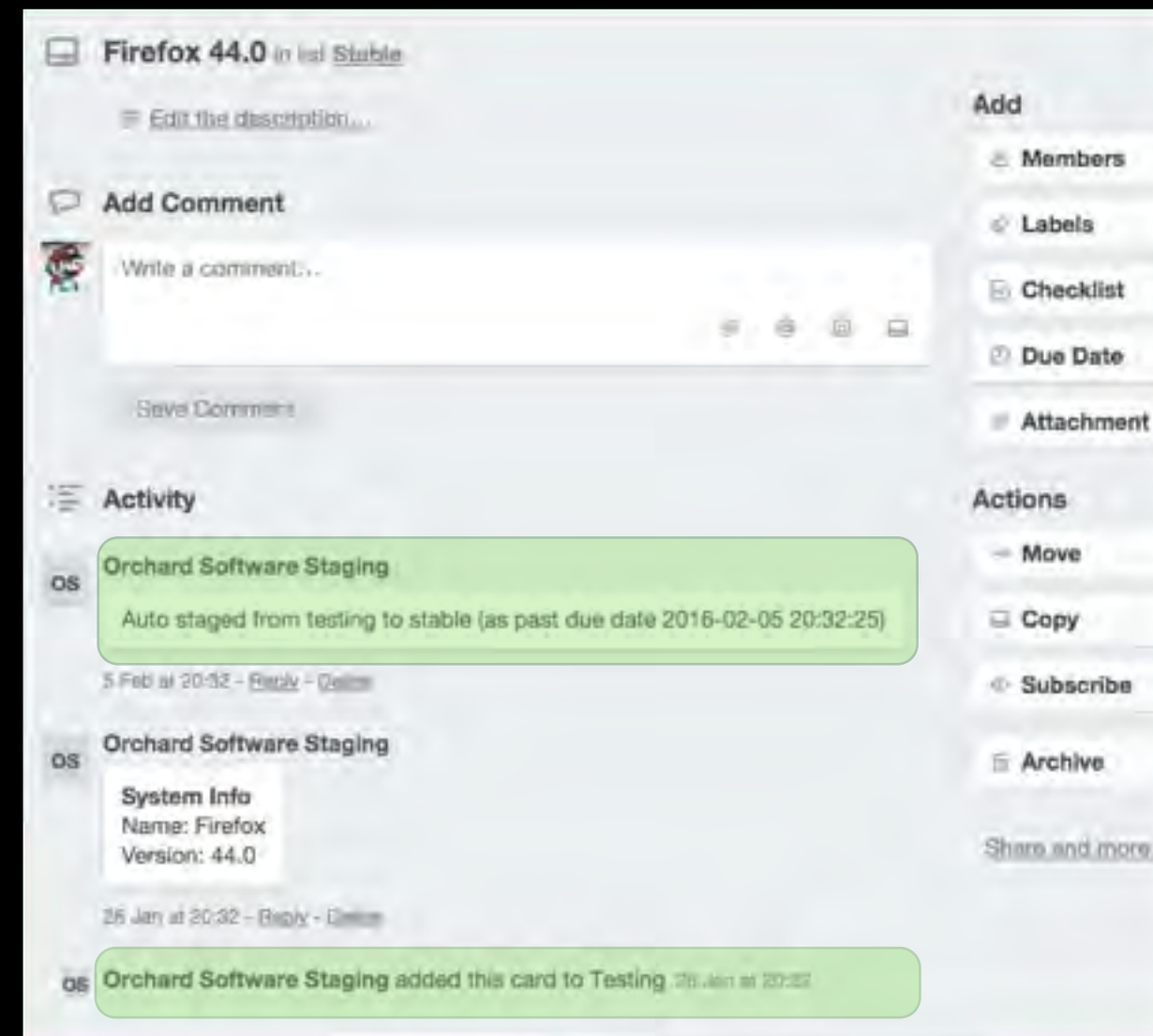


<https://github.com/ox-it/munki-staging>

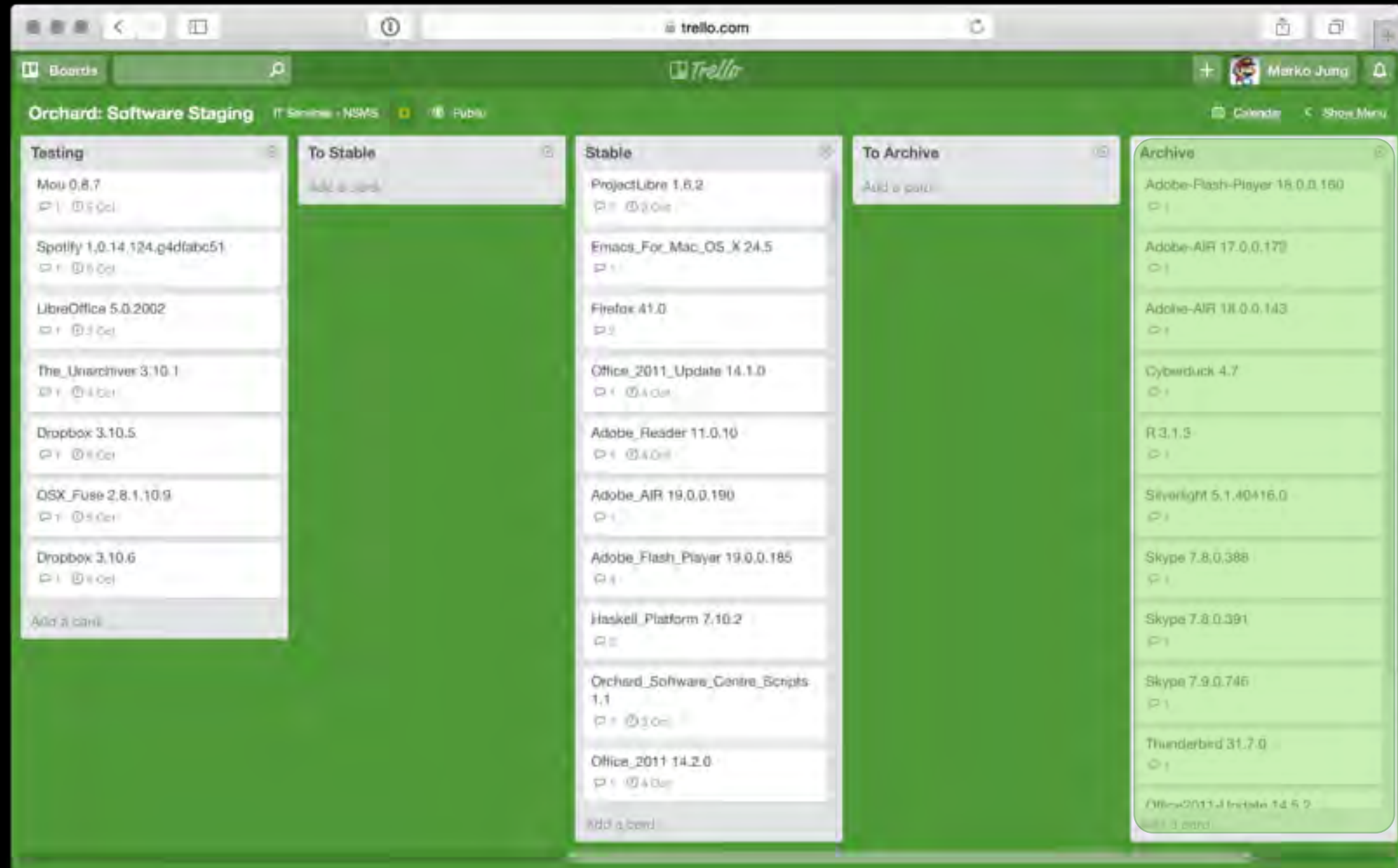
MUNKI-STAGING



- Rewrite of Grahan Gilbert's munki-trello
- New features:
 - Unlimited catalogs
 - Multiple munki repositories
 - Automated promotion
 - Schedule for auto promotion
 - RSS feed generation
 - Configuration file



MUNKI-STAGING



MUNKI-STAGING



1. Create a virtualenv including requirements,
\$ virtualenv munki-staging
\$ source munki-staging/bin/activate
\$ pip install trello
2. Ensure makecatalogs is present on your system (runs on Linux)
3. Clone or download muni-staging
\$ git clone https://github.com/ox-it/munki-staging.git
4. Create Trello APP key and set-up Trello user token
<https://trello.com/app-key>
<https://trello.com/docs/gettingstarted/#token>

MUNKI-STAGING



5. Write your configuration based on the provided template

```
# settings for a testing catalog and list
[munki_catalog_testing]
list=Testing
catalog=testing
stage_days=14
autostage=1
stage_to=production
stage_from=development
```

6. Run the script periodically (cron, launchd)

MUNKI-STAGING



Upcoming features:

- Package specific auto promote velocities
- Packages can obsolete older versions in catalogs

Known bugs:

- Dependency on a cloud service (Trello)
- ...

 <https://github.com/ox-it/munki-staging>



REBRANDING MUNKI



Two user facing management tools might be confusing:

1. ~~JAMF Self Service~~
Orchard Support Centre
2. ~~Managed Software Center~~
Orchard Software Centre

REBRANDING MUNKI



Customised build of the munkitools meta-package:

1. Clone git source
2. Replace strings, artwork, etc.
3. Add preflight and postflight scripts for munki
4. Add package postinstall script to configure munki
5. Use upstream build script to compile and package

 <https://github.com/ox-it/munki-rebrand>

Kudos to Arjen van Bochoven

<https://gist.github.com/bochoven/c1c656e0c2e1b1078dfd>

ENSURING JSS INVENTORY



1. Munki preflight script saves the modification time (mtime) of Munki install log

```
#!/bin/bash
```

```
DEFAULTS=/Library/Preferences/com.acme.lastinstalltime  
INSTALL_LOG_MTIME=$(stat -f "%m" "/private/var/log/munki/Install.log")  
defaults write "${DEFAULTS}" InstallLogMtime "${INSTALL_LOG_MTIME}"
```

2. Run managedsoftwareupdate operations

ENSURING JSS INVENTORY



3. Munki post flight executes `jamf recon` to update the computer inventory in the JSS iff there was a software change

```
#!/bin/bash
PATH=/usr/sbin:/usr/local/bin:/usr/local/jamf/bin:${PATH}
DEFAULTS=/Library/Preferences/com.acme.lastinstalltime
NEW_INSTALL_LOG_MTIME=$(stat -f "%m" /private/var/log/munki/Install.log)

if [ $(defaults read "${DEFAULTS}" InstallLogMtime) \
    -ne ${NEW_INSTALL_LOG_MTIME} ];
then
    jamf recon > /dev/null 2>&1 &
fi
```



PLEASE SEE MUNKI-REBRAND REPOSITORY FOR NICER VERSIONS OF THE SCRIPTS

ENSURING JSS INVENTORY



You have earned a trophy
Munki -> JSS reporting

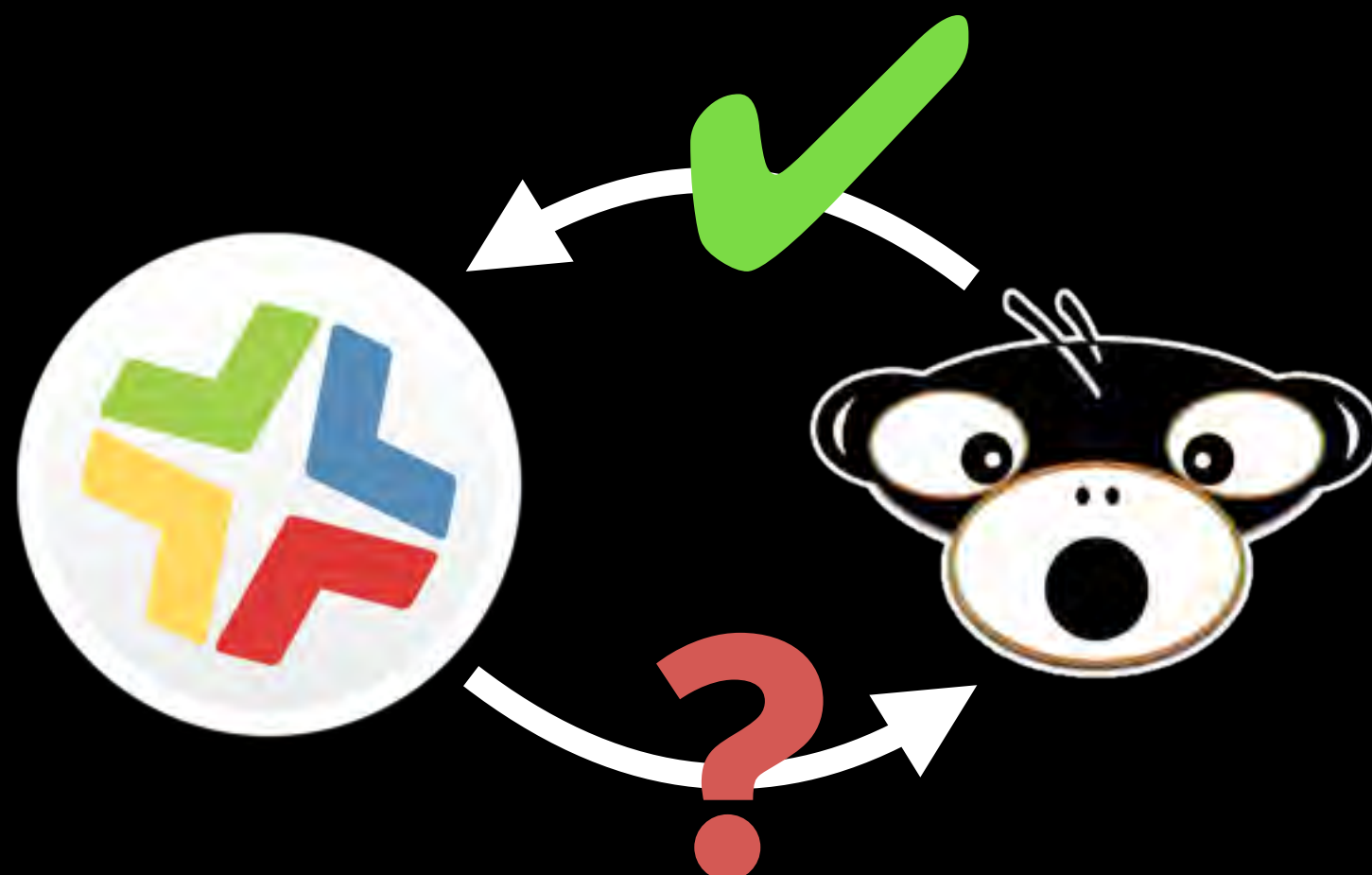
3. Munki post flight executes `jamf recon` to update the computer inventory in the JSS iff there was a software change

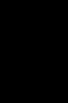
```
#!/bin/bash
PATH=/usr/sbin:/usr/local/bin:/usr/local/jamf/bin:${PATH}
DEFAULTS=/Library/Preferences/com.acme.lastinstalltime
NEW_INSTALL_LOG_MTIME=$(stat -f "%m" /private/var/log/munki/Install.log)

if [ $(defaults read "${DEFAULTS}" InstallLogMtime) \
    -ne ${NEW_INSTALL_LOG_MTIME} ];
then
    jamf recon > /dev/null 2>&1 &
fi
```



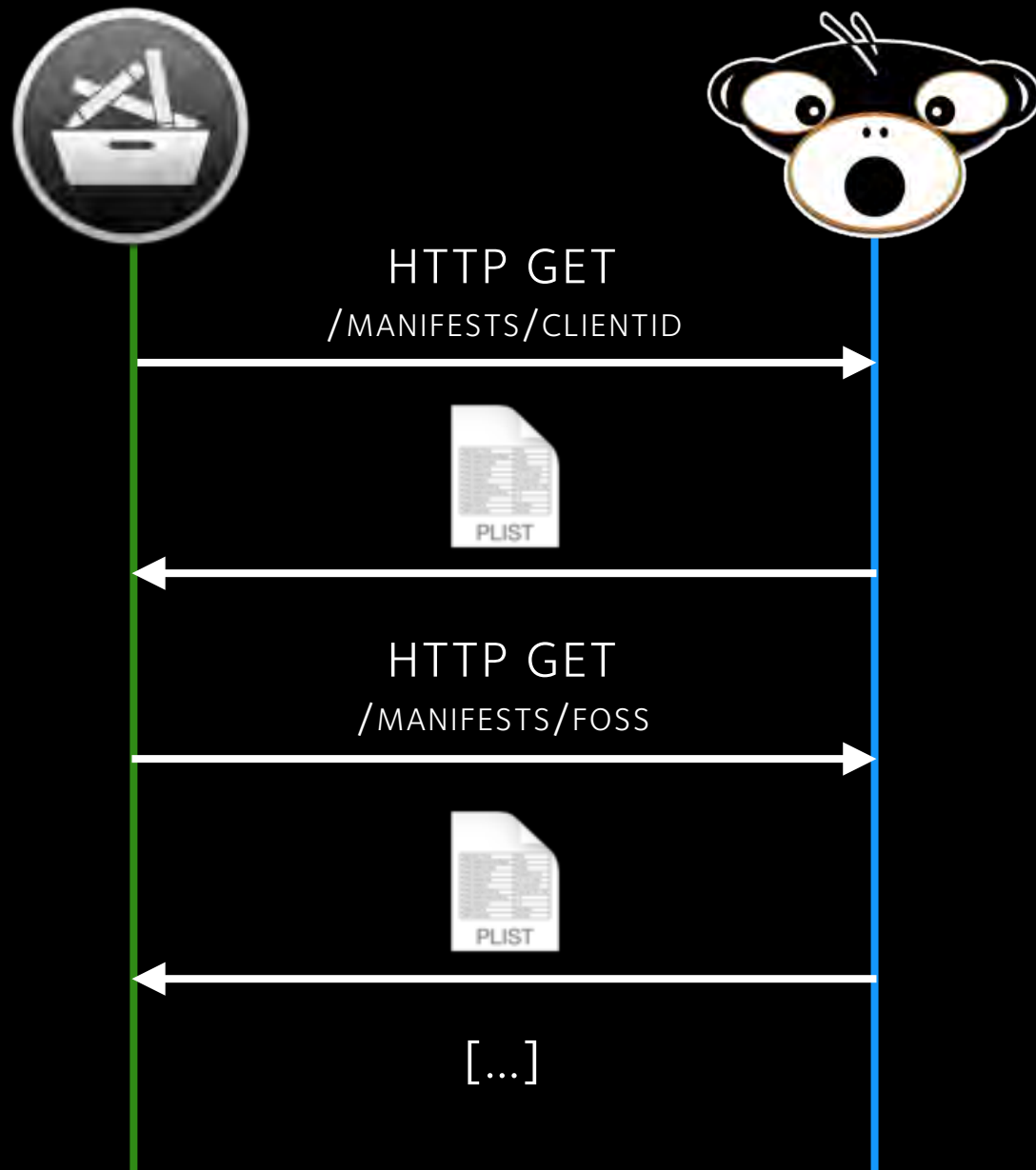
PLEASE SEE MUNKI-REBRAND REPOSITORY FOR NICER VERSIONS OF THE SCRIPTS





DEMO

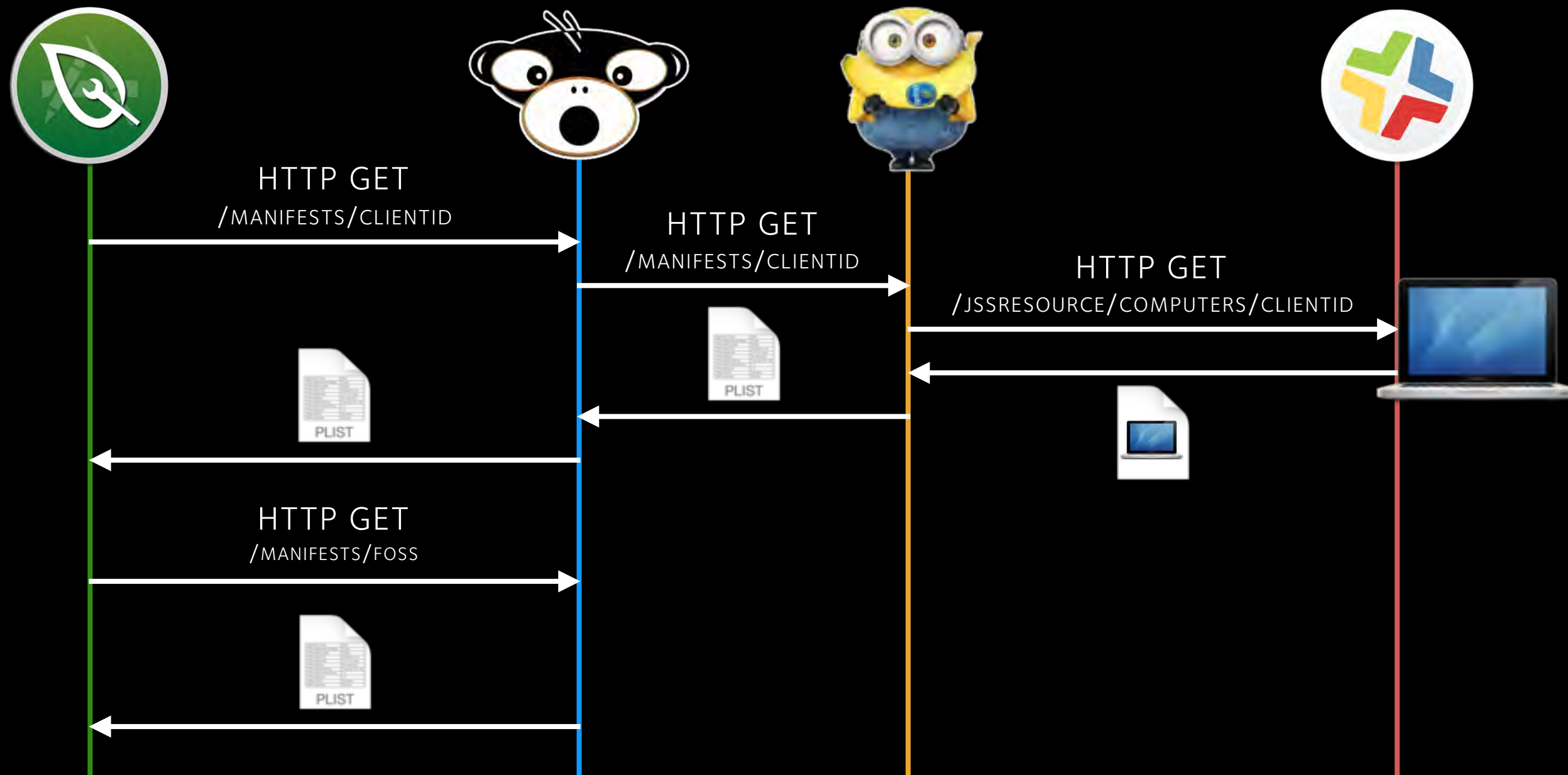
MESSAGE FLOW



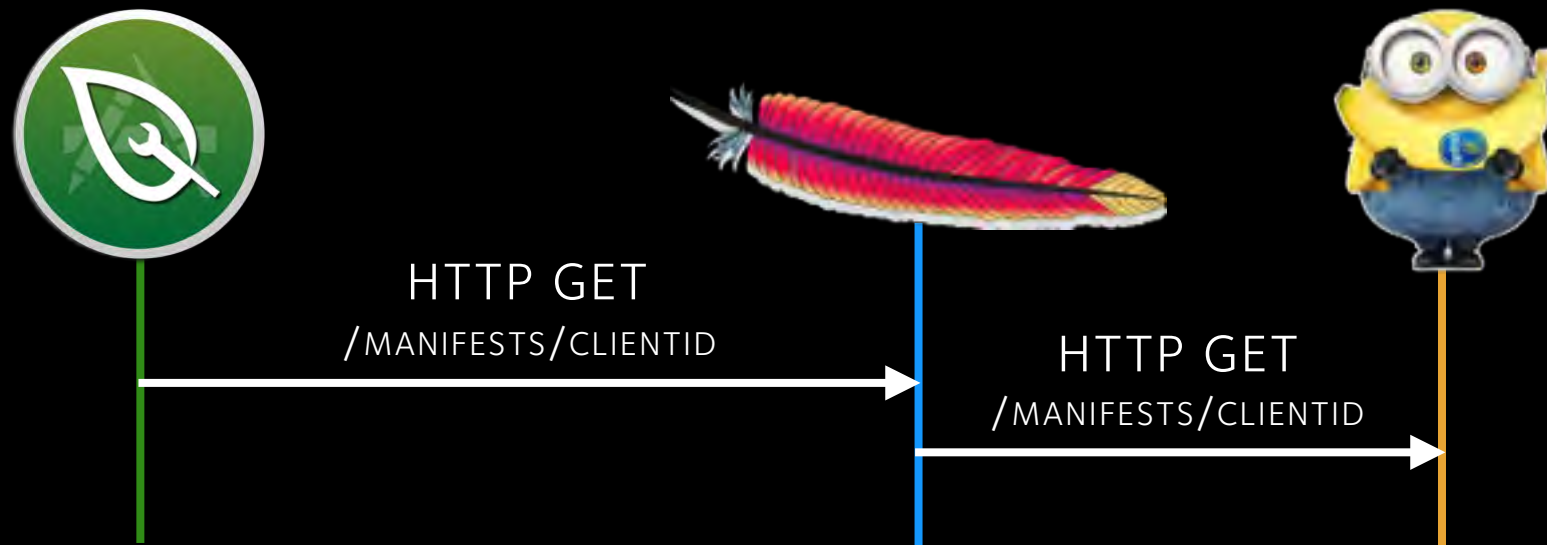
Example computer manifest

```
<dict>
  <key>catalogs</key>
  <array>
    <string>stable</string>
  </array>
  <key>included_manifests</key>
  <array>
    <string>foss</string>
  </array>
  <key>managed_installs</key>
  <array/>
  <key>managed_uninstalls</key>
  <array/>
  <key>managed_updates</key>
  <array/>
  <key>optional_installs</key>
  <array>
    <string>Firefox</string>
  </array>
</dict>
```


MESSAGE FLOW



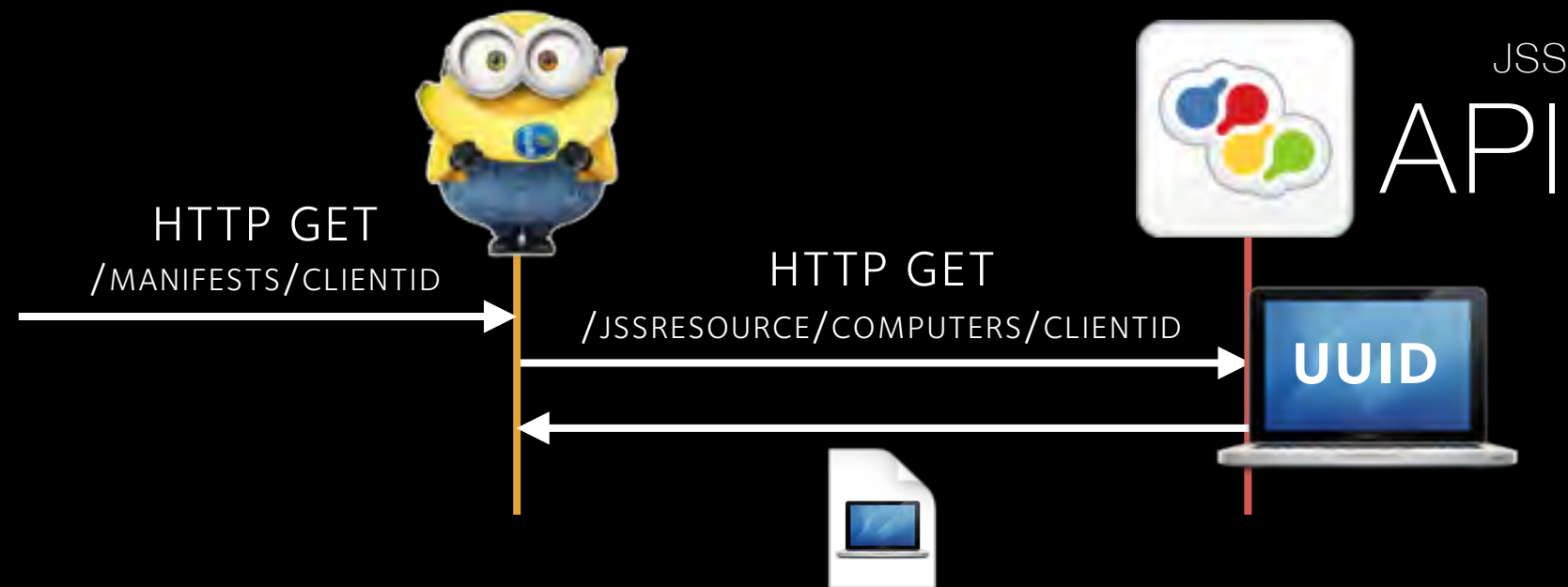
1. SERVE MANIFEST REQUESTS



- Munki repositories are flat files hosted on web servers
- Forward all requests for manifests to Manana:

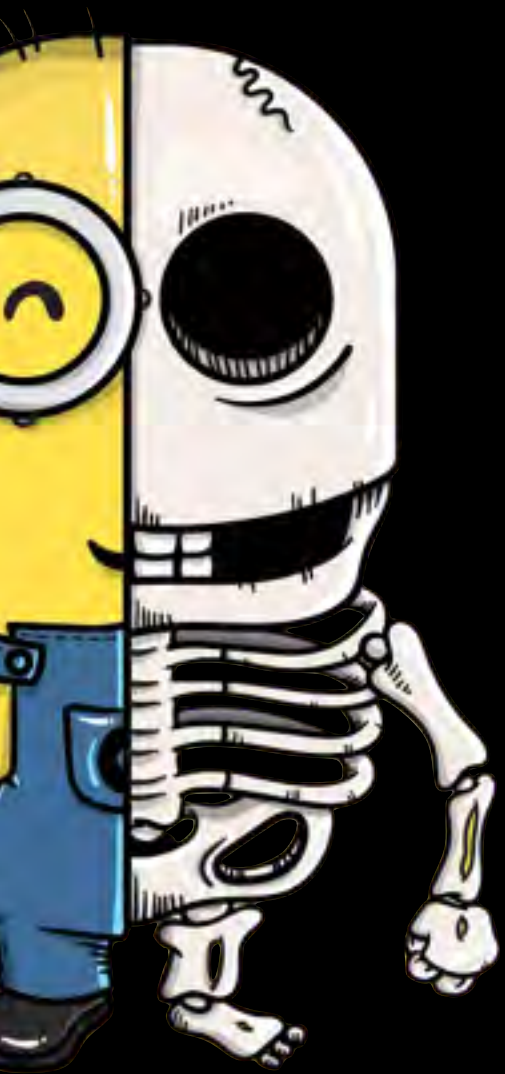
```
RewriteEngine On  
SSLProxyEngine On  
RewriteRule ^/production/manifests/(.*) \  
    https://manana.acme.com/jssmanifests/xml/$1 [P]
```

2. RETRIEVE JSS COMPUTER INFO



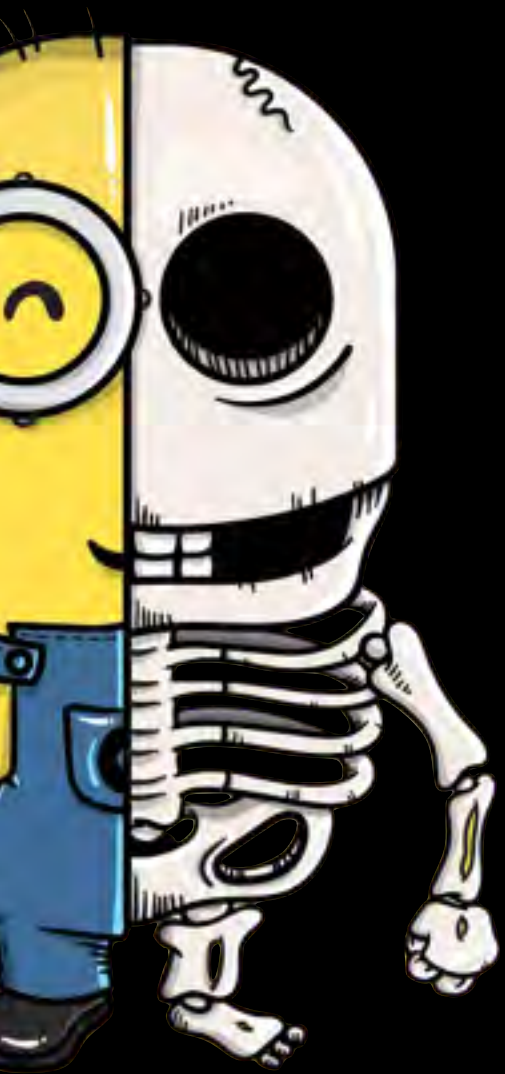
1. Unique CLIENTID required to identify a computer in the JSS
 2. UUID used to retrieve the computer inventory record via the JSS REST API
 3. JSS will return computer XML record
- UUID standardised format:
0F27B1D9-F915-5D5D-83A3-E07C57657ED9

3. DYNAMIC MANIFEST GENERATION



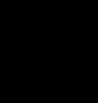
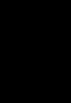
1. Download computer XML from JSS
2. Load computer manifest file from Munki repository (UUID or site_default)
3. Modify this computer manifest
4. Return dynamically generated manifest to web server (to be proxied to Munki client)

4. COMPUTER ATTRIBUTE MAPPINGS



- JSS computer XML fragment (XPath statement)
`DEPT = computer.xpath('//computer/location/department/text()')`
- Condition
`DEPT == 'wizardry'`
- Munki manifest addition / removal

```
<key>included_manifests</key>
<array>
  <string>book_of_spells</string>
</array>
<key>optional_installs</key>
<array>
  <string>abracadabra</string>
</array>
```




DEMO



MANANA AKA DYNAMIC MANIFESTS



- Use JSS Computer inventory information to dynamically create Munki manifests
 - match anything provided in the JSS API XML output
 - add or remove content to a per host or default template
 - New django App for MunkiWebAdmin
 - Merge with Steve Kueng's fork
 - Upgrade to django 1.8
 - Used in production environment
 - Actively maintained by Oxford Mac team
-  <http://github.com/ox-it/manana>

MANANA ROADMAP



- Support JSS Sites and JSS Users for
 - Object level permissions for Dynamic Manifest Mapping (view global mappings, create/change/delete site specific mappings)
 - Manifest permissions (view global manifests, create/change/delete site specific manifests)
- Caching
- Documentation
- (more) Unit Tests
- Shiny User Interface



Catalogs

Manifests

JSS Mappings

JSS Computer Attribute Mappings

	JSS Attribute Type	Value	Type	Priority	Actions
<input checked="" type="checkbox"/>	Department	Research & Development	foss	0	Edit Copy Delete
<input checked="" type="checkbox"/>	Extension Attribute	Device Ownership : ACME Inc.	office	0	Edit Copy Delete
<input checked="" type="checkbox"/>	Extension Attribute	Software Release : stable	stable	-1	Edit Copy Delete
<input checked="" type="checkbox"/>	Extension Attribute	Software Release : testing	stable	-1	Edit Copy Delete
<input checked="" type="checkbox"/>	Extension Attribute	Software Release : testing	testing	-2	Edit Copy Delete
<input checked="" type="checkbox"/>	Extension Attribute	Software Release : unstable	stable	-1	Edit Copy Delete
<input checked="" type="checkbox"/>	Extension Attribute	Software Release : unstable	testing	-2	Edit Copy Delete
<input checked="" type="checkbox"/>	Extension Attribute	Software Release : unstable	unstable	-3	Edit Copy Delete



- Authoritative Inventory
- MDM (Profiles, DEP)
- Volume Purchase Program (VPP)
- Orchard Support Centre (Self Service)



- Software deployment
- Orchard Software Centre





THANK YOU

MARKO JUNG

GALACTIC VICEROY OF RESEARCH EXCELLENCE

 m@mju.ng

 [@mjung](https://twitter.com/mjung)

 [fb.com/markohjung](https://facebook.com/markohjung)